NAG Toolbox

nag_sum_accelerate (c06ba)

1 Purpose

nag_sum_accelerate (c06ba) accelerates the convergence of a given convergent sequence to its limit.

2 Syntax

```
[ncall, result, abserr, work, ifail] = nag_sum_accelerate(seqn, ncall, work)
[ncall, result, abserr, work, ifail] = c06ba(seqn, ncall, work)
```

Note: the interface to this routine has changed since earlier releases of the toolbox:

At Mark 22: $lwork$ was removed from the interface.

3 Description

nag_sum_accelerate (c06ba) performs Shanks' transformation on a given sequence of real values by means of the Epsilon algorithm of Wynn (1956). A (possibly unreliable) estimate of the absolute error is also given.

The function must be called repetitively, once for each new term in the sequence.

4 References

Shanks D (1955) Nonlinear transformations of divergent and slowly convergent sequences *J. Math. Phys.* **34** 1–42

Wynn P (1956) On a device for computing the $e_m(S_n)$ transformation *Math. Tables Aids Comput.* **10** 91–96

5 Parameters

5.1 Compulsory Input Parameters

1: **seqn** – REAL (KIND=nag_wp)

The next term of the sequence to be considered.

2: **ncall** – INTEGER

On the first call **ncall** must be set to 0. Thereafter **ncall must not** be changed between calls.

3: **work**($lwork$) – REAL (KIND=nag_wp) array

$lwork$, the dimension of the array, must satisfy the constraint $lwork \geq 7$.

Used as workspace, but **must not** be changed between calls.

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **ncall** – INTEGER

The number of terms in the sequence that have been considered.

2: **result** – REAL (KIND=nag_wp)

The estimate of the limit of the sequence. For the first two calls, **result** $=$ **seqn**.

3: **abserr** – REAL (KIND=nag_wp)

An estimate of the absolute error in **result**. For the first three calls, **abserr** is set to a large machine-dependent constant.

4: **work**($lwork$) – REAL (KIND=nag_wp) array

5: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **ncall** < 0.

ifail $= 2$

On entry, $lwork < 7$.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The accuracy of the absolute error estimate **abserr** varies considerably with the type of sequence to which the function is applied. In general it is better when applied to oscillating sequences than to monotonic sequences where it may be a severe underestimate.

8 Further Comments

8.1 Timing

The time taken is approximately proportional to the final value of **ncall**.

8.2 Choice of $lwork$

For long sequences, a 'window' of the last n values can be used instead of all the terms of the sequence. Tests on a variety of problems indicate that a suitable value is $n = 50$; this implies a value for $lwork$ of 56. You are advised to experiment with other values for your own specific problems.

8.3 Convergence

nag_sum_accelerate (c06ba) will induce convergence in some divergent sequences. See Shanks (1955) for more details.

9 Example

This example attempts to sum the infinite series

$$\sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n^2} = \frac{\pi^2}{12}$$

by considering the sequence of partial sums

$$\sum_{n=1}^{1}, \sum_{n=1}^{2}, \sum_{n=1}^{3}, \ldots, \sum_{n=1}^{10}$$

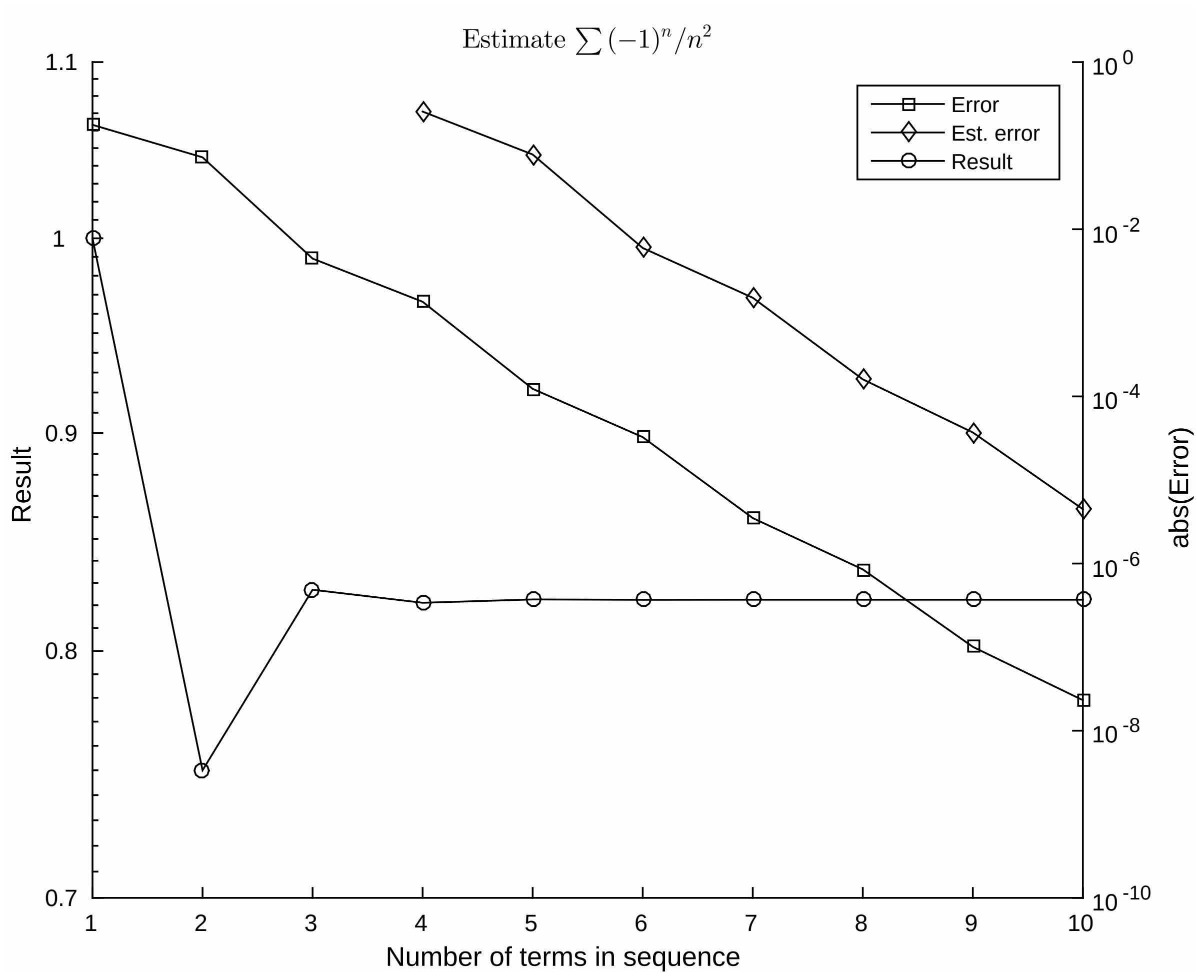
9.1 Program Text

```
     function c06ba_example

fprintf('c06ba example results\n\n');

% Set up initial values before calling NAG routine for the first time.
work = zeros(16, 1);
answer = pi^2/12.0;
ncall = nag_int(0);
sig = 1.0;
seqn = 0.0;

% Initialize arrays for plotting.
x = zeros(1,10);
seqnArr = zeros(1,10);
resArr = zeros(1,10);
errArr = zeros(1,10);
absArr = zeros(1,10);

% Output headings.
disp('No. of  Term   Estimate  Estimated    Actual');
disp(' terms  value  of limit  abs error     error');

% Loop over terms.
for i = 1:10;
  r = double(i);
  seqn = seqn + sig/(r^2);

  % NB - input and output ncall *must* be the same variable (not e.g.
  % ncallOut).  Ditto for work.
  [ncall, result, abserr, work, ifail] =  ...
  c06ba(seqn, ncall, work);
  err = result-answer;
  sig = -sig;
  if i <= 3
    % First three calls of c06ba return no error estimate.
    fprintf('%4d %8.4f %8.4f      -      %11.2e\n', i, seqn, result, err);
  else
    fprintf('%4d %8.4f %8.4f %11.2e %11.2e\n', i, seqn, result, abserr, err);
  end

  % Accumulate results for plotting.
  x(i) = i;
  seqnArr(i) = seqn;
  resArr(i) = result;
  errArr(i) = err;
  if i > 3
    absArr(i) = abserr;
  end
```

```
end

% Plot results.
fig1 = figure;
display_plot(x, resArr, errArr, absArr);

function display_plot(x, y1, y2, y3)

  % Use a log plot for both curves.
  [haxes, hline1, hline2] = plotyy(x, y1, x, abs(y2), 'semilogy', 'semilogy');
  % Set the axis limits and the tick specifications to beautify the plot.
  set(haxes(1), 'YLim', [0.7     1.1]);
  set(haxes(2), 'YLim', [1.0e-10 1]);
  set(haxes(1), 'YMinorTick', 'on');
  set(haxes(2), 'YMinorTick', 'on');
  set(haxes(1), 'YTick', [0.7:0.1:1.1]);
  set(haxes(2), 'YTick', [1.0e-10 1.0e-8 1.0e-6 1.0e-4 1.0e-2 1]);
  % Specify this labels explicitly.
  set(haxes(1), 'YTickLabel', [0.7; 0.8; 0.9; 1.0; 1.1]);
  for iaxis = 1:2
    % These properties must be the same for both sets of axes.
    set(haxes(iaxis), 'XLim', [1 10]);
    set(haxes(iaxis), 'XTick', [1:10]);
    set(haxes(iaxis), 'Position',...
        [0.13 0.0910780669144981 0.715317725752508 0.802973977695167]);
  end
  set(gca, 'box', 'off'); % no ticks on opposite axes.
  % Set the title.
  t1 = '{Estimate $\sum';
  t2 = '{\displaystyle (-1)^n}/{\displaystyle n^2}$}';
  t1 = strcat(t1,t2);
  title(t1,'Interpreter','latex');
  % Label the axes.
  xlabel('Number of terms in sequence');
  ylabel(haxes(1),'Result');
  ylabel(haxes(2),'abs(Error)');

  % Add the third curve (still a log plot).
  axes(haxes(2));
  hold on;
  hline3 = plot(x(4:10), y3(4:10));
  % Set some features of the three lines.
  set(hline1, 'Linewidth', 0.5, 'Marker', 'o');
  set(hline2, 'Linewidth', 0.5, 'Marker', 's');
  set(hline3, 'Linewidth', 0.5, 'Marker', 'd','Color','Magenta');
  % Add legend.
  legend( 'Error', 'Est. error', 'Result', 'Location', 'NorthEast');
```

9.2 Program Results

```
     c06ba example results

No. of  Term   Estimate  Estimated    Actual
 terms  value  of limit  abs error     error
    1   1.0000   1.0000      -         1.78e-01
    2   0.7500   0.7500      -        -7.25e-02
    3   0.8611   0.8269      -         4.46e-03
    4   0.7986   0.8211    2.56e-01   -1.36e-03
    5   0.8386   0.8226    7.84e-02    1.23e-04
    6   0.8108   0.8224    5.97e-03   -3.26e-05
    7   0.8312   0.8225    1.52e-03    3.50e-06
    8   0.8156   0.8225    1.60e-04   -8.51e-07
    9   0.8280   0.8225    3.70e-05    1.01e-07
   10   0.8180   0.8225    4.48e-06   -2.32e-08
```


Estimate $\sum(-1)^n/n^2$
Error
Est. error
Result
1.1
1
0.9
0.8
0.7
Result
10^0
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
abs(Error)
1 2 3 4 5 6 7 8 9 10
Number of terms in sequence